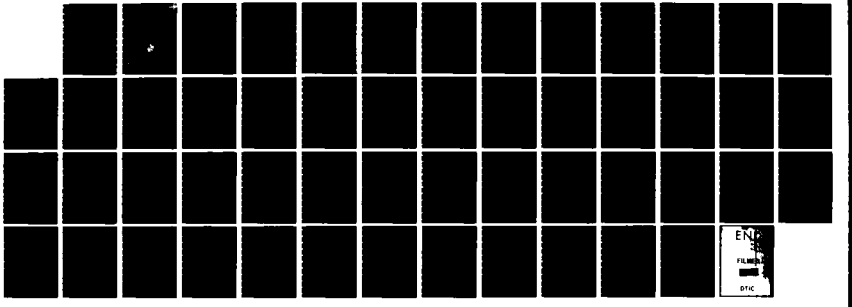
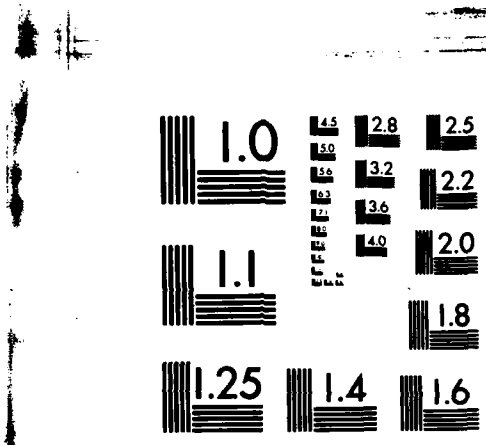
AD-A137 473 GENIE: A COMPUTER-BASED TASK FOR EXPERIMENTS IN HUMAN-COMPUTER INTERACTIO..(U) VIRGINIA POLYTECHNIC INST AND STATE UNIV BLACKSBURG COMPUTER S.. 1/1

UNCLASSIFIED T E LINDQUIST ET AL. OCT 83 CSIE-83-10 F/G 5/8 NL





(2)

GENIE: A COMPUTER-BASED TASK FOR EXPERIMENTS IN HUMAN-COMPUTER INTERACTION

T. E. Lindquist
R. G. Fainter
M. T. Hakkinen
S. R. Guy
J. F. Maynard

Virginia Polytechnic Institute
and State University
Computer Science
Industrial Engineering and Operations Research
BLACKSBURG, VIRGINIA 24061

CSIE-83-10

October 1983

2

GENIE: A COMPUTER-BASED TASK FOR EXPERIMENTS IN HUMAN-COMPUTER INTERACTION

T. E. Lindquist
R. G. Fainter
M. T. Hakkinen
S. R. Guy
J. F. Maynard

TECHNICAL REPORT

Prepared for
Engineering Psychology Group Office of Naval Research
ONR Contract Number N00014-81-K-0143
Work Unit Number NR SRO-101

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: CSIE-83-10	2. GOVT ACCESSION NO.: A137473
3. RECIPIENT'S CATALOG NUMBER:	
4. TITLE (and Subtitle): GENIE: A COMPUTER-BASED TASK FOR EXPERIMENTS IN HUMAN-COMPUTER INTERACTION	5. TYPE OF REPORT & PERIOD COVERED: Technical
	6. PERFORMING ORG. REPORT NUMBER:
7. AUTHOR(s): T.E. Lindquist, R. G. Fainter, M.T. Hakkinen, S.R. Guy and J.F. Maynard	8. CONTRACT OR GRANT NUMBER(s): N00014-81-K-0143
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Computer Science, Virginia Polytechnic Institute & State University, Blacksburg, VA 24061	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: 61153N42; RR04209; RR0420901; NR SR0-101
11. CONTROLLING OFFICE NAME AND ADDRESS: Office of Naval Research, Code 442, 800 North Quincy Street, Arlington, VA 22217	12. REPORT DATE: October 1983
	13. NUMBER OF PAGES: 40
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)	15. SECURITY CLASS. (of this report): Unclassified
	15a. DECLASSIFICATION DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Human Computer Interfaces, Compiler-Compiler
User Interface Experimentation, Task Environment

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The results of many human-computer interaction studies are often not generalizable because the task environment in which they are run does not possess characteristics common to other interfaces. In this paper we describe a generalized task environment that is directly applicable to several interesting real-world tasks, and that contains elements appearing in almost every system having a human-computer interface. The environment is implemented through a software system called GENIE (Generic ENvironment for Interactive

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

Experiments), and is based on controlling the motion of vehicles through three-dimensional space. Aside from providing a task with common characteristics, GENIE's implementation was designed to allow for adaptation to a variety of studies. The user's interface to the system has been constructed in such a way as to minimize the effort necessary for change.

The paper first describes the development of the GENIE software system and then presents its structure. The user's view of the system is discussed followed by a presentation of the facilities available to the experimenter. Software components of the system are described from a functional level, and finally, three example experiments that use the system are described.

ACKNOWLEDGEMENTS

This research was supported by the Office of Naval Research under ONR Contract Number N00014-81-K-0143, and Work Unit Number NR SRO-101. The effort was supported by the Engineering Psychology Group Office of Naval Research, under the technical direction of Dr. John J. O'Hare.

ABSTRACT

The results of many human-computer interaction studies are often not generalizable because the task environment in which they are run does not possess characteristics common to other interfaces. In this paper we describe a generalized task environment that is directly applicable to several interesting real-world tasks, and that contains elements appearing in almost every system having a human-computer interface. The environment is implemented through a software system called GENIE (Generic ENvironment for Interactive Experiments), and is based on controlling the motion of vehicles through three dimensional space. Aside from providing a task with common characteristics, GENIE's implementation was designed to allow for adaptation to a variety of studies. The user's interface to the system has been constructed in such a way as to minimize the effort necessary for change.

The paper first describes the development of the GENIE software system and then presents its structure. The user's view of the system is discussed followed by a presentation of the facilities available to the experimenter. Software components of the system are described from a functional level, and finally, three example experiments that use the system are described.

CONTENTS

I. PROBLEM STATEMENT

Studies on human-computer interactions commonly present a problem for the user to solve using a computerized task environment. By varying certain factors in the task, users are exposed to different experimental conditions. Measurements are then taken on specific aspects of user performance. At least two alternatives exist in the design of such a task environment. One approach uses a specific existing task; for example, a study of how a factor alters the performance of a participant in an editing task, or how certain factors change performance when using a specific operating system's command language like the UNIX shell. On the other hand, a generic or nondescript environment possessing characteristics that typify many human-computer interactions may be designed. In the first case results obtained are applicable only to the specific task. Strictly speaking, the results obtained in the second case are only applicable in the generic environment, but it is generally accepted that these results can be extrapolated to specific tasks having the same characteristics. GENIE (Generic ENvironment for Interactive Experiments), the test-bed presented in this paper, provides a generic task flexible enough to be used in several human-computer interaction studies. It possesses characteristics that are applicable

to many specific computer tasks. In this test-bed, major known (and hypothesized) effects and principles governing human-computer communication can be tested to determine, for example, whether dominance, negative interference, amplification, or other interactions occur in conjunction with other major effects and principles. The extent to which results can be extrapolated from the GENIE task to other human-computer interactions remains to be seen.

The basic form of GENIE assumes an air traffic control task in which human-computer dialogue takes place between the user, acting as a controller, and GENIE's airplane logic, acting as the pilot. By the use of various commands the user directs the flight of one or more airplanes. There are any number of real-world tasks that are analogous to the one GENIE presents including:

1. air traffic control,
2. movement of ships, or
3. docking of satellites or other spacecraft with an orbiting space station.

If altitude is ignored, the problem reverts from three-space to two-space and an entirely new variety of applications arise; for example:

1. refueling operations of ships at sea,
2. movement of trains in a railyard,
3. trucks arriving at a warehouse,
4. distribution of goods in a warehouse, or
5. movement of automobiles on a city's streets.

GENIE can be made to appear as any of these tasks by changing the display that GENIE presents to the user and the language that is used. The underlying functionality of the system, however, would remain constant. In each of the instances cited above, GENIE would play the part of one or more vehicle operators, while the participant using GENIE would be the agency controlling the vehicles.

GENIE has been designed to possess many of the same characteristics that exist in a variety of real-world human-computer tasks. These characteristics include user computation, deep dialogues (dialogues that require many interdependent interactions), user searches, system failures, various information sources for the user, and varying degrees of user workload. For example, the dialogue that takes place between the user and GENIE is characterized as deep since many commands are often necessary to control an aircraft on a given flight path. Each command given to GENIE must be based upon information gained in all preceding commands. The ability of the user to employ previous information and to estimate the needed heading changes determines the number of interactions required. As another example, an experimenter can alter participant workload by predefining the number of aircraft being controlled at a given time or by modifying the relative speeds of the aircraft.

GENIE is designed to be flexible in that it can be configured to accommodate various task configurations. Output from the system may appear on a color graphics terminal for one user, but may be voice to another. Also, the form of input and output may vary for different participants in that one may use menu driven commands while another uses parametric command entry. Further, the task environment can be configured to use spatial representations of information, and can vary the amount of displayed information. This high degree of flexibility is clearly important, and the ease with which the flexibility can be exercised by the experimenter is a major advantage of GENIE.

The measurement of user performance on GENIE is accomplished by data collection or metering. Performance measurement in any human/computer task can be both internal and external to the computer-based task. For example, direct observation of the participant by the experimenter is sometimes necessary to observe head movements toward multiple displays, or to record referrals to hardcopy manuals or help information. These measurements are typically hand-scored and entered into a computer system for later analysis. Internal metering, on the other hand, provides precise "time stamps" indicating when various events and commands are processed. The experimenter can be provided with the facility to collect response times; for example, the time interval between the first occurrence of

an aircraft emergency and the participant's response to the emergency. Other examples of data that can be internally metered by GENIE include the type and frequency of commands used on an overall, or per aircraft basis, the time required to land a given aircraft, number of command entry errors, and the number of times a user requests information from the system.

II. USER VIEW OF GENIE

GENIE has two human interfaces: one that the participant uses to perform an experimental task and another that the experimenter uses to configure a task. The participant's interface to the system consists of two graphics displays and a set of commands for communicating with the airplanes being controlled. Feedback to th participant is possible by different media including voice, text, and graphics. Although the participant's view of GENIE is elaborate, the experimenter's interface to the system is terse. Through a file of commands that can be reused, the experimenter sets parameters defining specific tasks. Each of these user interfaces is presented in this section.

A. THE USER'S COMMAND LANGUAGE

To the participant, Genie appears as a generalized air traffic control task. The participant acts as the controller by guiding one or more aircraft through approach and landing patterns. The specific environment described is

that used by Hakkinen and Williges as just one example of how GENIE might look to the user. It is presented through a discussion of the input and output media, the controlling task that the user is to perform, and finally, the commands available to the user for performing the task.

1. The Controlling Task

Keyboard. The user is seated at a Digital Equipment Corporation (DEC) GIGI graphics keyboard. All commands are entered by the user through the keyboard one line per command. The alphabetic, numeric, space, delete-line, backspace, and return keys are the only functional keys in this environment.

Displays. Two BARCO GD33 color monitors are placed side-by-side at the user workstation, which is shown in Figure 1. The display at the right side is primary and represents a radar screen and auxiliary information areas. The radar display depicts a square area 80.5 km (50 miles) wide, and the crossed circle at the left center of the screen is the airport. A line leading from the crossed circle is the final-approach segment, which the aircraft must follow to land. The dotted line depicts the approach pattern the aircraft should follow to the final-approach segment.

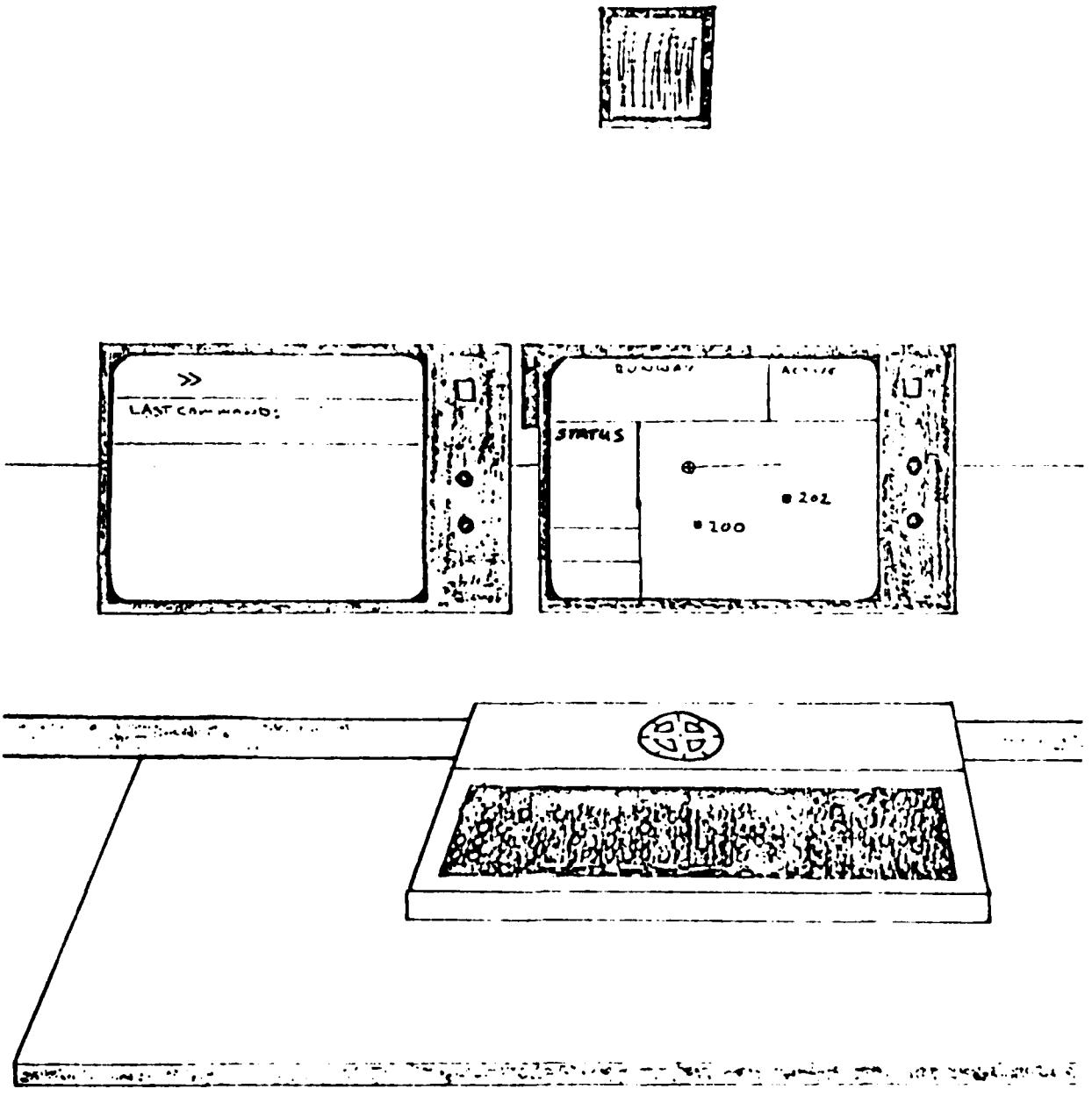


Figure 1. The User's Workstation.

Commands that the user enters appear on the top line of the display at the left side. When the "RETURN" key is pressed, the command moves down to the "LAST COMMAND" area, and the top line clears. If the command contains an error, the message "COMMAND ERROR" appears beneath the "LAST COMMAND" line and two short beeps are generated by the keyboard unit. To aid the user in guiding the aircraft, it is sometimes necessary to obtain information that is not available on the radar display. The "AIRCRAFT INFORMATION" area of the display on the right side is used to present this type of data.

Landing aircraft. As an air traffic controller, the user navigates aircraft around the display area using a specialized vocabulary to communicate information between the controller, the aircraft, and the computer system. The aircraft, symbolized on the display by a small white block and an adjacent three-digit identification number, first appears on the lower edge of the display. As each aircraft enters the control area, its call sign appears in the "ACTIVE AIRCRAFT" list on the "AIRCRAFT INFORMATION" area of the primary display. Scores indicating the number of planes landed and the number of missed approaches are tabulated on the primary display.

One feature of this task is that all directions (such as north, west, south, and east) are designated by their

compass headings. A simple vocabulary for travel in this environment has been defined based upon compass headings. Each turn that must be made is specified by both the new compass heading and the direction of turn. The alternative directions, left and right, are relative to the vehicle's current direction of travel.

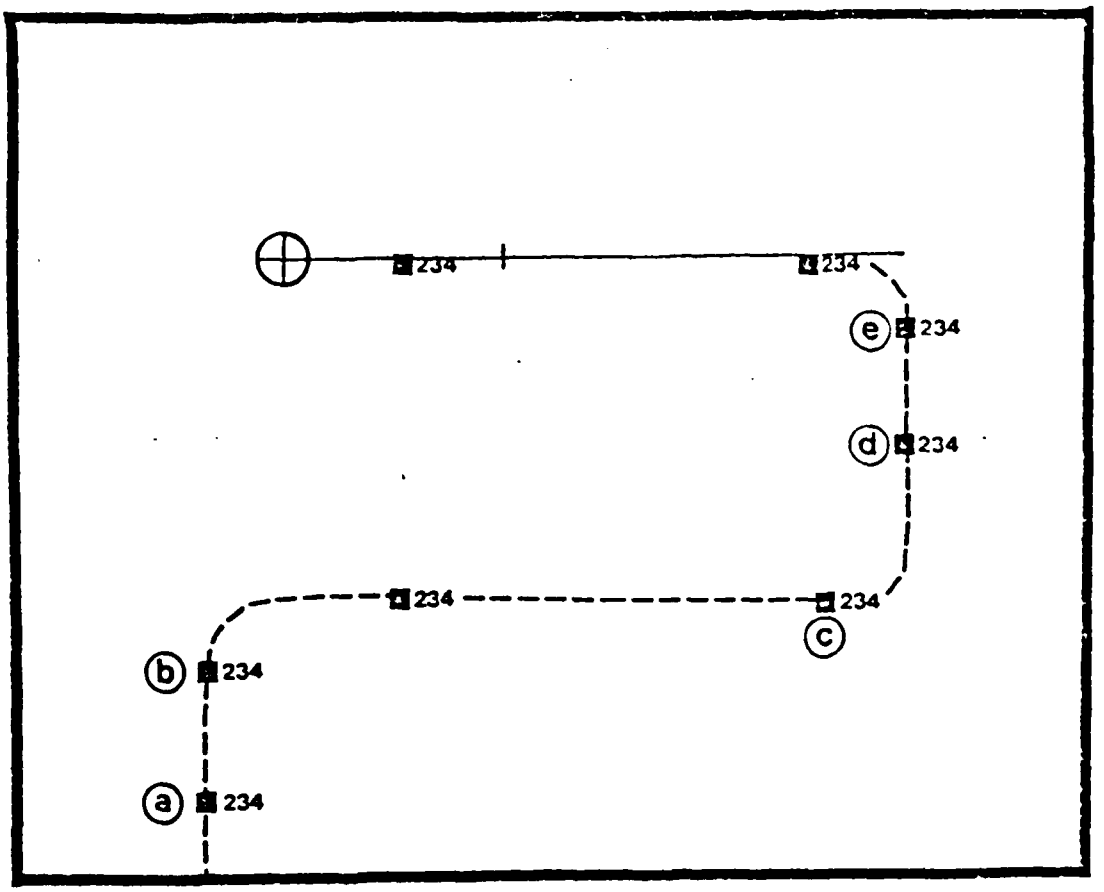
Aircraft enter the control area from the south travelling north 6.28 radians (360 degrees). When an aircraft enters, it is travelling at its cruising speed of 2778 km/hr (1500 knots). but it is helpful to command the aircraft to reduce its speed as soon as possible to ease the controlling task. The original heading of the aircraft would lead it approximately to the airport without further guidance by the participant. According to GENIE rules of airspace usage, however, each aircraft must be directed to follow a standard path leading to the east 1.6 radians (90 degrees) and then turning to the north 6.28 radians (360 degrees). The aircraft flies north until it is almost in line with the final segment of the approach pattern.

The final-approach segment is represented by the solid- and dotted-white line pointing to the right from the airport symbol. As the aircraft reaches the vicinity of the final-approach sgment, it should be directed to once again reduce speed, and then turn in toward the airport. To fly down the center line of the final-approach segment, the aircraft must

have a heading of 4.7 radians (270 degrees). If the heading and radial differ by more than one or two degrees, the aircraft will begin to drift away from the final-approach course. Valid final-approach headings to the airport are in the range of 4.6 to 4.9 radians (260 to 280 degrees). If the aircraft's heading is outside this range, it will not land. If the aircraft is on the final-approach radial, but then leaves, the aircraft disappears from the screen and is counted as a missed approach.

Once an aircraft has been given a command to turn to the final-approach segment, heading information should be requested from the aircraft to insure that it is within the bounds of the final-approach course. The show command should be used to determine whether the aircraft is on final approach, and if there is a radial/heading difference. If there is a deviation, the direction the plane must turn to correct for it will be given. This information appears on the "LANDING INFORMATION" list in the "AIRCRAFT INFORMATION" area of the primary display. The aircraft's heading, deviation, and corrective turn (if needed) are only shown for aircraft on final approach up to 74 km (40 nmi) east from the airport.

Figure 2 shows the radar display and the necessary commands for guiding a single aircraft (identification number 234) to an airport landing. Under optimal



ⓐ 234 speed 1200

ⓑ 234 turn right heading 090

ⓒ 234 turn left heading 360

ⓓ 234 speed 600

ⓔ 234 turn left heading 270

Figure 2. Radar display and sample GENIE commands.

conditions, only five commands would be needed to direct the aircraft to the landing. These commands would be needed to direct the aircraft to the landing. These commands are shown in the figure. Should a delay occur in transmitting a command or should an erroneous command be transmitted, additional commands may be needed to return an aircraft to the correct flight path.

One point that complicates the control of aircraft is that the distance required to complete a turn is proportional to its speed. The higher the aircraft's speed, the wider the turn that it makes. A certain amount of familiarity with the rate at which an airplane turns is needed for initiating turns at the proper time.

2. Command Language Summary The syntactic form and complete description of the commands that a user may enter are presented below.

SPEED

ID_NUMBER speed N

The speed command specifies which aircraft is to have its speed changed by providing an aircraft "id" number. Following this identifier, the command name, speed, and a value in knots (N) was entered. Valid speeds are in the range or 741 to 2778 km/hr (400 to 1500 knots). Cruising speed is 2778 km/hr (1500 knots); approach speed should be

2222 km/hr (1200 knots); and speed on final-approach should be 1111 km/hr (600 knots). An example of the command directing aircraft 150 to change its speed to 1111 km/hr (600 knots) is:

150 speed 600

TURN

ID_NUMBER turn DIRECTION heading DEGREES

The turn command requires specification of which aircraft is to be turned, the direction (left or right), and the new heading. The new heading of the aircraft was specified in degrees, and valid values are 0-6.8 radians (000 through 360 de grees). All three digits of the new heading, including leading zeros, must be specified. To turn aircraft 200 left to a heading of 1.6 radians (90 degrees) the controller would type:

200 turn left heading 090

SHOW

show ID_NUMBER

To obtain landing information about an aircraft, the user enters the show command. Responses to the command appear on the primary display in the "AIRCRAFT INFORMATION" area listed under "LANDING INFORMATION." This information is

automatically updated and displayed until a new show command is issued for another aircraft. To obtain landing information about aircraft 300 the controller would type:

show 300

SAY

ID_NUMBER say INFORMATION

With the say command, information about the aircraft's speed, fuel status, radial, or heading can be requested. Responses to the say command are displayed in the "AIRCRAFT INFORMATION" area on the primary display. The appropriate say command is elected by first typing the ID_NUMBER, and then pressing one of the special say command function keys on the top row of the keypad. No 'RETURN' key is pressed for this command, and 'BACKSPACE' and 'DELETE' can only be used to correct the ID_NUMBER. An example of each use of the command is shown below:

200 say speed

202 say fuel status

300 say heading

304 say radial

REPLY

reply FUEL-WEIGHT

At various times during a trial, the GENIE system needs to know the fuel status of certain aircraft. GENIE requests this information of the controller who in turn uses the say command to obtain the fuel weight from the vehicle (ID_NUMBER say fuel status). The controller then communicates the weight in pounds (eg 25493) to GENIE using the reply command. This type of interaction exemplifies those tasks in which two human-computer interfaces are needed to solve a problem. In this case, the controller interfaces with both the airplane and the GENIE system. An example of the command is:

reply 25493

ACK

ID_NUMBER ack EMERGENCY TYPE

When an emergency occurs, the response should be made as quickly as possible with the ack ack (acknowledge) command. When this command is issued GENIE does not generate any response. For example:

202 ack engine on fire

B. THE EXPERIMENTER'S COMMAND LANGUAGE

The experimenter controls the actions that take place during a trial via the experimenter's command language. Specifying when and what airplanes are depicted, what emergencies occur, and what information is displayed is all done through the experimenter's command language. In effect, the experimenter writes a GENIE "program" or profile, which specifies what actions are to take place and the time for the execution of these actions. Seven commands can be used to construct such a program: DEFINE, BEGIN, CREATE, EMERGENCY, SPEED, and HALT.

DEFINE. The DEFINE command is used to establish the initial configuration of GENIE for a given trial. Each DEFINE command may have one of five parameters, and several DEFINE commands may be used. The operands were:

AIRPORT LOCATION

REFRESH RATE

SCREEN DIMENSION

TAG

APPROACH HEADING

AIRPORT LOCATION determines whether the airport symbol is located in the center of the radar display area or at the

center of a quadrant. The parameter REFRESH RATE is the number of seconds that elapse between each update of the screen. The parameter SCREEN DIMENSION is a number that defines the distance from the left edge of the radar display to its right edge in nautical miles. The TAG option of the DEFINE command indicates what information is to be displayed in the textual tag beside each airplane on the screen. After the word TAG, one or more of the words CALL SIGN, AIRSPEED, or ALTITUDE may appear. The APPROACH HEADING parameter defines the final-approach bearing that the aircraft must fly to the landing.

BEGIN. The BEGIN command specifies the time at which GENIE is to draw the screen and begin processing the experimenter's action commands. The BEGIN must follow the definition commands and must precede the action commands (CREATE, EMERGENCY, SPEED, and HALT).

CREATE. The CREATE command is an instruction to depict a new aircraft on the display. The experimenter specifies, in the following order, after the word CREATE, the type of aircraft (XTF-1, XTF-2, XTF-3, XTB-1, RCU-1), the number of aircraft in the flight (1 to 4), the call sign of the aircraft, and its location (bearing and distance) relative to the airport. Examples of this command appear at the end of this section.

EMERGENCY. The EMERGENCY command conveys that an aircraft has some trouble that requires special handling. The keyword EMERGENCY is followed by the call sign of the aircraft. The valid emergencies are HYDRAULIC, FIRE, COMM TRANSMITTER, COMM RECEIVER, TACAN, ADF, IFFSIF, BATTLE DAMAGE, GYRO, SLATS, AILERONS, RUDDER, ELEVATOR or LANDING GEAR. Only one of these may occur on an emergency command, but several emergency commands may refer to a single aircraft. If the keyword CANCEL follows the call sign, then the specified emergency no longer applies.

SPEED. The SPEED command indicates that the chosen aircraft is to accelerate or decelerate to the given speed. The acceleration used for this speed change is 3.7 km/s. The keyword SPEED is followed by the call sign and the new speed.

HALT. The HALT command terminates the trial. It has no operands.

The experimenter's language interpreter reads commands from a disk file which is created in advance by the the experimenter. Each command is made up of two parts:

the elapsed time after the beginning of the trial indicating when the command is to be executed, and

the command itself, described in the preceding several paragraphs.

For example, if the record

```
0730 create xtf-1 1 122 180 20
```

appeared in the file, then seven minutes thirty seconds into the trial (i.e., after the BEGIN is executed) this create command will be executed, causing one aircraft of type xtf-1 to be depicted, with call sign 122 and located 3.2 radians (180 degrees) from the landing strip at 37 km (20 nmi).

There is a structure that must be observed in the experiment profile. All define commands must be placed together at the beginning of the file. These must be followed by one BEGIN command. A sample GENIE profile is shown below for a 20 minute session during which several aircraft are depicted and two emergencies are processed.

```
0000 define refresh rate 3
0000 define screen dimension 50
0000 define tag call sign
0000 define airport location ul
0000 define approach heading 315
0000 begin
0015 create xtf-1 1 110 180 20
0245 create xtf-2 1 211 045 30
0315 create xtf-1 1 111 180 25
0320 emergency 211 hydraulic
0330 emergency 211 landing gear
0500 create xtf-3 1 344 090 50
1000 emergency 211 cancel landing gear
2000 halt
```

III. SOFTWARE DESCRIPTION

The operation of GENIE from the controller's and experimenter's view was described in Section II and in this section the software structure for that system is outlined. This is done at the module/process level by first describing the interactions between modules and then presenting the modules individually. A more detailed description of the GENIE software system in the form of a software maintenance manual has been reported in [2]. GENIE is organized into five processes named:

GNESUBJ (The user's command interpreting process)

GNEEXPR (The experimenter's command interpreting process)

GNETIME (A synchronizing process)

GNEDBAS (The vehicle database process)

GNEDISP (The display managing process)

Figure 3 depicts the interprocess communication and synchronization links among GENIE modules. The user's process, GNESUBJ, recognizes the user's commands, and like all other processes is a DEC VAX 11/780 Pascal program. The experimenter's process, GNEEXPR, interprets and executes the experimenter's commands. It obtains input from a disk file

containing a profile of trials. Execution of the first define command in the profile causes the database process, GNEDBAS, to be created, and execution of the BEGIN command causes the synchronizing process to be created. Once this timer is initiated, it requests, periodically thereafter, that the database process produce new locations for each airplane currently active. The new location of an airplane is based on elapsed time and speed. As each new location is determined, the database requests that the display process, GNEDISP, write a blip on the screen in the calculated position. After recognizing and analyzing each command (experimenter's or controller's), the database process is requested to make the appropriate changes. All five of these processes run until the experimenter's profile issues the HALT command; at that time, the experimenter's process terminates and causes all other processes to be terminated.

A. GNESUBJ AND GNEEXPR: COMMAND LANGUAGE IMPLEMENTATION

Any command to GENIE, whether from the controller or the experimenter, undergoes three distinct phases of analysis in its processing. First, the words, or lexical units, are isolated and collapsed to a usable code; second,

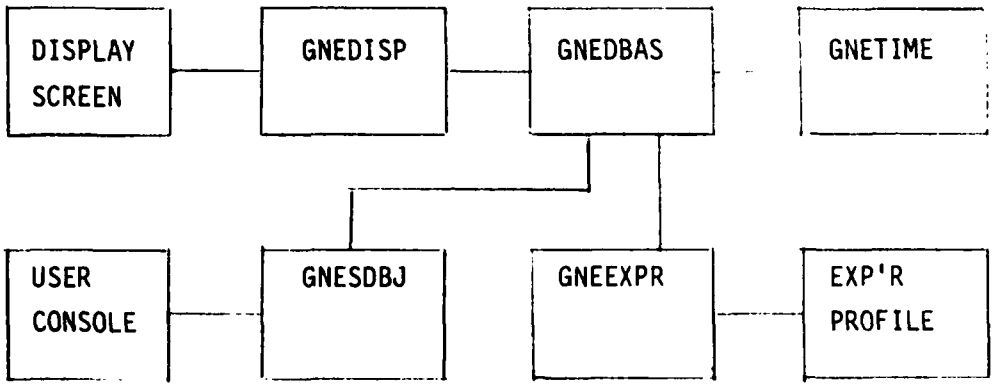


Figure 3. Steady-state macro structure of GENIE.

the syntactic structure of the command with its arguments is detailed; and finally, any processing necessary to carry out the command is performed. To provide for flexibility of command structure, a compiler-compiler performs these three phases of processing. Based on grammatical specifications of the commands, the compiler-compiler produces a set of tables. The tables are then linked with a grammar independent driver routine, a lexical analyzer, and semantic routines to form the program. The compiler-compiler selected, DEC LLPARS [4], performs a table-driven top-down analysis of commands. LLPARS consists of a table-builder and a driver routine, LLDRV. The table-builder accepts as

input an augmented LL(1) grammar. This grammar describes the syntax of the commands and also names semantic action routines to be called as various commands and command fragments are recognized.

With the employment of a table-driven language-processing system like LLPARS, only the following components need be provided to implement or change the GENIE command structure:

A BNF-like specification of the command language,

An input scan function named LLSCAN, and

The semantic action routines specified by the command grammar.

Each of these components is discussed in the following sections. More detailed information on the compiler-compiler LLPARS may be found in [4].

1. BNF-Like Specification of the Language

The user language and the experimenter language are described by separate augmented LL(1) grammars. A more thorough treatment of LL(1) grammars and, more generally, the use of BNF (Backus-Naur Form) or context-free grammars

are provided in [5]. For LLPARS, a language specification consists of three major sections: the terminal definitions, the keyword-definitions, and the productions.

The terminal definition section defines, in part, the tokens or lexical elements used in the grammar. For example, both the user and experimenter languages define error, end-of-line (EOL), and a numeric digit as tokens. Although these terminals have no semantic values, they are the elements used to build a command sentence.

The keyword-definition section of the grammar includes all command words of the language. For example, the user's language includes the words "ALTITUDE", "SPEED", and "TURN", for directing aircraft. Similarly, the experimenter's language employs the words "BEGIN", "CANCEL", "CREATE", and "EMERGENCY", for establishing the air traffic control scenario for a given experiment. Each keyword is a lexical element of the language, and in the same way as a terminal symbol, each keyword is reduced to a token value. This value is returned by the scanner when the keyword is encountered in the input stream. For example, in parsing the user command:

100 say speed

the scanner first returns the value assigned to a digit three times (for 1,0, and 0), and then returns the values

assigned to the keywords "SAY" and "SPEED" respectively. On the next call, LLSCAN returns the end-of-line token.

The production section is the heart of the grammar since it contains the list of rules which define the syntax of commands. These productions also specify the semantic action routines to be called as various command fragments are recognized. The productions appear in a BNF-like form in which syntactic categories are defined by a sequence of lexical elements (keywords and terminals) and syntactic categories (non-terminals). For example, the following production is the first (highest level) production for the user's language.

```
AIR-CONTROLLER      = EOL {PROMPT} COMMAND AIR-CONTROLLER
                    | ERROR {ERRMSG} EOL AIR-CONTROLLER
```

These rules are interpreted as definitions in which the syntactic category on the left-hand-side of the '=' is defined by the right-hand-side. Vertical bars, '|', on the right-hand-side, separate alternative definitions. The AIR-CONTROLLER rule states that on its first call from the parser, the scanner should return the end-of-line token (EOL). In other words, initially there is no command to be parsed. This condition triggers a call to the semantic action routine, PROMPT, to prompt the controller for a command. The word "COMMAND" in this production is a

syntactic category defined in subsequent productions and describes the various user commands. Thus, the command received by the PROMPT routine from the controller should be one of the commands defined by the syntactic category "COMMAND". The last component on this line is the syntactic category "AIR-CONTROLLER". Since this is also the syntactic category being defined, the operation is recursive. When one command has been parsed, the production is repeated: the EOL token is returned by the scanner, PROMPT is called to get another command from the controller, and the new command is parsed. Thus, once initiated, the parsing operation continues until parsing is stopped by execution of the HALT instruction.

2. The Input Scan Function and Semantic Action Routines

The scanner, LLSCAN, is a FORTRAN function called by the parser. The function analyzes the first non-blank character or sequence of characters in the command input stream and returns a token value indicating the type of symbol found. The scanner is written so as to identify and return the tokens as defined in the terminal and keyword sections of the grammar.

The semantic action routines are named in productions of the language specification by enclosing them in braces, '{' and '}'. These routines are called by the parser, LLDRV, as various commands or command fragments are processed. Based on the type of operation performed, the semantic routines are classified into general semantic-action routines and command-specific ones. The general group of semantic-action routines are used in both the user's and the experimenter's processes to aid in parsing commands. The routines ERRMSG, NUMERIC, PROMPT, and HALT form this group. Command-specific routines gather and code information that is to be conveyed to the database process. Many of the command-specific routines perform limited semantic checks prior to passing information to the database process.

B. GNETIME: THE SYNCHRONIZING PROCESS

The synchronizing process is to trigger the database process periodically and to cause it to produce a new configuration of the airplanes. It does this by calling a software service "long_delay" with the number of seconds between traversals. Upon returning from "long_delay", the

timer requests the database to perform the update operation. The timer then waits for a message indicating that the update has occurred before returning to hibernation. This process is repeated as long as GENIE is active.

C. GNEDBAS: THE DATABASE PROCESS

The database process creates and maintains an ordered binary tree that stores information about each airplane. It performs a traversal of this tree when requested to do so by the timer. During the traversal, each airplane's position is updated, and a request is made to have the display modified by GNEDISP. Additions to the tree are made as required through the experimenter's language processor. Nodes are automatically deleted when the vehicle that the node represents gets within a prespecified distance of the destination on final approach.

During the updating traversal, each node is sent to the "change" procedure. This procedure calls several other procedures which actually modify the node. Procedure "new_pos" computes the new (x,y) position on the screen and the new radial position with respect to the destination. After the new position of the aircraft has been computed,

its angular motion is calculated by the procedure "turning". Aircraft turn at a rate of .05 radians/s (3 degrees per second) until they are within .2 radian (10 degrees) of the assigned heading. In this procedure, the difference between the current vehicle heading and the assigned heading is computed. If this difference is greater than the amount that the vehicle will turn in the next traversal, then the turn continues and an intermediate heading is calculated. If the vehicle is within .2 radian (10 degrees) of the assigned heading, the rate of turn is reduced by one-half. The procedure "change" is next invoked to decide if the aircraft is in a position from which a landing may be made. (The aircraft must be on final approach and within 18 km (10 nmi) of the destination.) If this is the case, the "prepare_to_land" procedure lowers the landing gear, slowing the aircraft to approach speed and begins the aircraft's descent to the destination. This is called putting the aircraft in "landing configuration." If, at any time after the aircraft begins its final approach to the destination, it departs from the final approach course, the "go_around" procedure is executed. This causes the aircraft to climb back to 1.5 km (5000 feet) altitude and accelerate to cruise speed.

D. GNEDISP: THE DISPLAY PROCESS

The GNEDISP process draws the screen image and places blips representing each airplane on the screen. It makes use of a version of the CORE graphics system [1]. After being invoked, initializations are performed that include drawing all areas on the screen, drawing the airport, and filling in the status area. Once initializations have been completed, the process enters a wait-for-request loop. Requests are made by the database process for such activities as updating the position of an existing aircraft, creating an image for a new aircraft, removing the image of inactive aircraft, and modifying various data areas on the screen. GNEDISP is a single PASCAL program which is comprised of five routines named:

ACTIVE
CARRIER
PLANE
REFRESH
SCREEN

ACTIVE maintains the list of the active aircraft. The structure of this list is a tree with each entry having several attributes. CARRIER is the routine which draws the airport where the experimenter wants it to be placed. PLANE draws and/or erases the aircraft from the screen, and it writes information adjacent to the aircraft. The information may be any combination of call sign, altitude,

or air speed, as selected by the experimenter. REFRESH is called when the controller requests information on a particular airplane. The information displayed includes the call sign, heading, fuel status, landing gear position, altitude, distance to the destination, bearing to the destination, estimated time of arrival at the destination, the communication frequency currently being used, and remarks about the airplane. Finally, SCREEN draws the boxes which make up the screen, and it also writes the heading information into all of the boxes.

IV. EXPERIMENTS BASED ON GENIE

To date, GENIE has been adapted for use in three human-computer interaction experiments. The two experiments described below deal with task allocation (allowing the computer to perform certain duties otherwise considered part of the operator's responsibility) and voice output. Rieger and Greenstein [6] describe the task-allocation study in more detail, and Hakkinen and Williges [3] detail the voice output study. Each study used a specifically modified version of GENIE that accommodated the particular needs of that experiment. Although the results of the experiments are not of importance here, the two experiments are described to illustrate how GENIE may be used.

A. TASK ALLOCATION

A version of GENIE has been used to study the effects of human-computer task allocation. In this version of GENIE, the experimenter selected an error threshold. When this threshold was reached, the computer was made available for relieving the user of some controlling duties. The

duration for which the operator may make use of this reallocation was also controlled by the experimenter. The commands specifically associated with the task-allocation configuration were as follows:

TAKE	[something]	
something	NEXT	[how many]
	QUAD	[which ones]
	EMERG	
	DISPLAY	
	[call signs]	
how many	digit [0-9]	
which ones	digit [,digit [,digit]]	
call signs	side_number [,side_number]*	

The TAKE NEXT command turns control of the next n aircraft over to the computer. The value of n, in the range of zero to nine, is specified by the operator. The TAKE QUAD command turns control of up to three quadrants of the screen to the computer. The TAKE EMERG command allows the computer to control all aircraft which are experiencing emergency situations, and the TAKE DISPLAY command gives the computer control of all aircraft which have gone beyond the range of the screen. The "TAKE call signs" command instructs the computer to control aircraft which are specified in the command line.

The experimenter's language was extended to facilitate the above commands. There was a need to be able to set the initial aircraft entry point and the spacing between points. This was accomplished by extending the range of the DEFINE command. Four keywords were added: INITIAL, INTERVAL, THRESHOLD, and DELTA (DELTA defines the amount of time during which the operator may issue one of the TAKE commands).

Task allocation became active when the number of errors encountered exceeded a threshold. Two types of errors were defined to determine the threshold. The first was a command error. In this type, the operator had entered an invalid or unknown command; this included misspellings. The second was an action error. This included aircraft flying off the screen, missed approaches, and collisions of vehicles. Once the threshold had been reached, the error counter was reset to zero. In this manner, the operator was able to invoke the task allocation strategy numerous times.

B. VOICE OUTPUT

Hakkinen and Williges [3] examined synthesized-speech presentation of information using the GENIE environment.

The purpose of one of their studies was to examine the issue of whether alerting cues are needed before the presentation of critical voice messages, and whether the amount of information presented by the synthesizer has any effect on this need.

Two levels were considered for the synthesized-speech message environment. In the first level, only emergency messages were presented by the synthesizer, with other less urgent information appearing on either of the two GENIE visual displays. In the second level, messages which had appeared on the visual displays were instead presented via the voice synthesizer. These messages included a vocal response to the SAY command, an announcement that a new aircraft had entered the control area, and a periodic request by the system for fuel-status information on a given aircraft.

Modifications to the GENIE task provided an environment suitable for these studies. The modified task required the operator to use the SPEED and TURN commands to control aircraft entering the display screen and to guide them along a predefined approach pattern to landing. Aircraft emergencies and fuel-status requests occurred at various points during an experimental session. SHOW and SAY commands were utilized by the operator to check on aircraft status and adherence to the approach pattern.

The approach to implementing the voice synthesizer, a Federal Screw Works Votrax ML-1, was to incorporate the synthesizer drivers directly into the GENIE processes. The experimenter process, GNEEXPR, was the site of emergency and fuel-status-request message generation. A new experimenter command, TALK, provided the capability for generating low priority messages (such as the fuel-status request) and high priority messages. When an emergency message was activated by the experimenter, a check was made to determine whether an alerting cue is to be presented before the emergency message. If so, a modified TELERAY terminal generated a tone and light cue. At the experimenter's option, low priority messages were presented either via the voice synthesizer or a visual display.

Within the user process, GNESUBJ, voice messages were generated by the SAY command. Four types of information could be presented: aircraft heading, speed, radial, and fuel status. A check was made by the INFOIN subroutine to determine whether the responses to the SAY command should be presented by the voice synthesizer or displayed visually.

Four function keys on the DEC GIGI keypad were programmed to generate the SAY commands. Using this method, the operator has only to key in an aircraft's three-digit call sign and then press the appropriate function key to obtain the desired information.

The primary data collected from the environment was to determine how quickly an operator responded to voice messages, and how accurately such messages were transcribed. Accurate timing of the operator's keying activity was obtained by modifying the GNESUBJ PROMPT routine to provide metering of individual keystrokes. Each operator keypress received a time stamp that was then stored in a disk file for later analysis. A typical 48-minute session produced about 4000 keypresses. Also collected were measures of aircraft performance parameters at various points in the approach pattern.

Early pretesting indicated that operators would perform better if scoring information were made available on the display screen. This resulted in modifications to the GNEDISP process to record the number of aircraft landed and missed, and then display these counts in a score-board on the primary display.

Training people to serve as controllers in the GENIE environment did not prove difficult. An 18-page instruction guide was utilized in conjunction with a 45-minute training session. Controllers were successfully trained during a single one and a half-hour session.

V. SUMMARY

The design and implementation of a computerized solution to any problem requiring a human-computer interface is a difficult challenge to do correctly. Ideally, the problem of designing a human-computer interface involves applying a set of validated human-interaction principles. The principles are applied in consideration of the task being solved, the users of the system, and the environment in which the system is to execute. The result of applying these principles is an optimal human-computer interface which must be implemented within the pragmatics of the application. The pragmatics may determine that the resources required for the human interface are not available, or they may determine that the computational requirements to support the interface are excessive. Although this type of pragmatic tradeoff must always be made, it is important to balance pragmatics with the human-computer principles determining the form of the user interface.

Unfortunately, the necessary principles are not sufficiently complete. One reason for the lack of principles is that they must often be established through extensive experimentation. In part, the extensive experimentation is a result of using diverse test-beds to

examine different aspects of the same principle. It is this problem that the GENIE system addresses by providing a highly-flexible task environment. While flexibility is important, a generic test-bed must also be rich in the characteristics of dialogue required by the task. In designing GENIE, the characteristics of varying degrees of workload, highly interdependent dialogues, multi-device interactions, and graphic as well as textually-oriented interactions were all basic criteria. Thus, GENIE is a system which is both flexible and applicable to many specific human-computer interaction tasks.

VI. REFERENCES

[1] Ehrich,R.W. "A Two-Dimensional Core Graphics System for Research in Human-Computer Interfaces," Technical Report CSIE-81-4, Virginia Tech Departments of Computer Science and IEOR, Blacksburg, VA 24061, October, 1981.

[2] Fainter,R.G.; et.al. "GENIE: A Software Description of the GENIE System," Technical Report Department of Computer Science, Virginia Tech, Blacksburg, VA 24061, (in preparation).

[3] Hakkinen,M.T., and Williges,B.H. "Synthesized Voice Warning Messages: Effects of Alerting Cues and Message Environment," Human Factors Society Meeting, Seattle Washington, October, 1982.

[4] Morse,J.A. "LLPARS Users Manual," Digital Equipment Corporation Technical Report DEC/TR-90, Corporate Research Group, Maynard MA,01754,

[5] Pyster,A. Compiler Design and Construction, Van Nostrand Reinhold, New York, 1981.

[6] Reiger, C.A. and Greenstein,J.S., "The effects of Dialogue-Based Task Allocation on System Performance in a Computer-Aided Air Traffic Control Task," in Behavior Research Methods and Instrumentation (in press).

October 1983

OFFICE OF NAVAL RESEARCH

Engineering Psychology Group

TECHNICAL REPORTS DISTRIBUTION LIST

OSD

CAPT Paul R. Chatelier
Office of the Deputy Under Secretary
of Defense
OUSDRE (E&LS)
Pentagon, Room 3D129
Washington, D.C. 20301

Dr. Dennis Leedom
Office of the Deputy Under Secretary
of Defense (C^3I)
Pentagon
Washington, D.C. 20301

Department of the Navy

Engineering Psychology Group
Office of Naval Research
Code 442 EP
Arlington, VA 22217 (2 cys.)

Communication & Computer Technology
Programs
Code 240
Office of Naval Research
800 North Quincy Street
Arlington, VA 22217

Manpower, Personnel & Training
Programs
Code 270
Office of Naval Research
800 North Quincy Street
Arlington, VA 22217

Information Sciences Division
Code 433
Office of Naval Research
800 North Quincy Street
Arlington, VA 22217

Special Assistant for Marine Corps
Matters
Code 100M
Office of Naval Research
800 North Quincy Street
Arlington, VA 22217

Department of the Navy

CDR James Offutt, Officer-in-Charge
ONR Detachment
1030 East Green Street
Pasadena, CA 91106

Director
Naval Research Laboratory
Technical Information Division
Code 2627
Washington, D.C. 20375

Dr. Michael Melich
Communications Sciences Division
Code 7500
Naval Research Laboratory
Washington, D.C. 20375

Dr. Robert E. Conley
Office of Chief of Naval Operations
Command and Control
OP-094H
Washington, D.C. 20350

Dr. Robert G. Smith
Office of the Chief of Naval
Operations, OP987H
Personnel Logistics Plans
Washington, D.C. 20350

Combat Control Systems Department
Code 35
Naval Underwater Systems Center
Newport, RI 02840

Human Factors Department
Code N-71
Naval Training Equipment Center
Orlando, FL 32813

Dr. Alfred F. Smode
Training Analysis and Evaluation
Group
Orlando, FL 32813

Department of the Navy

CDR Norman E. Lane
Code N-7A
Naval Training Equipment Center
Orlando, FL 32813

Dr. Gary Poock
Operations Research Department
Naval Postgraduate School
Monterey, CA 98940

Dean of Research Administration
Naval Postgraduate School
Monterey, CA 93940

Mr. John Impagliazzo
Code 101
Naval Underwater Systems Center
Newport, RI 02840

Dr. A.L. Slafkosky
Scientific Advisor
Commandant of the Marine Corps
Code RD-1
Washington, D.C. 20380

Dr. L. Chmura
Naval Research Laboratory
Code 7592
Computer Sciences & Systems
Washington, D.C. 20375

Chief, C^3 Division
Development Center
MCDEC
Quantico, VA 22134

Human Factors Technology Administrator
Office of Naval Technology
Code MAT 0722
800 N. Quincy Street
Arlington, VA 22217

Commander
Naval Air Systems Command
Human Factors Programs
NAVAIR 334A
Washington, D.C. 20361

Commander
Naval Air Systems Command
Crew Station Design
NAVAIR 5313
Washington, D.C. 20361

Department of the Navy

Mr. Philip Andrews
Naval Sea Systems Command
NAVSEA 03416
Washington, D.C. 20362

Commander
Naval Electronics Systems Command
Human Factors Engineering Branch
Code 81323
Washington, D.C. 20360

Larry Olmstead
Naval Surface Weapons Center
NSWC/DL
Code N-32
Dahlgren, VA 22448

Dr. George Moeller
Human Factors Engineering Branch
Submarine Medical Research Lab
Naval Submarine Base
Groton, CT 06340

Commander, Naval Air Force,
U.S. Pacific Fleet
ATTN: Dr. James McGrath
Naval Air Station, North Island
San Diego, CA 92135

Navy Personnel Research and
Development Center
Planning & Appraisal Division
San Diego, CA 92152

Dr. Robert Blanchard
Navy Personnel Research and
Development Center
Command and Support Systems
San Diego, CA 92152

CDR J. Funaro
Human Factors Engineering Division
Naval Air Development Center
Warminster, PA 18974

Mr. Stephen Merriman
Human Factors Engineering Division
Naval Air Development Center
Warminster, PA 18974

Mr. Jeffrey Grossman
Human Factors Branch
Code 3152
Naval Weapons Center
China Lake, CA 93555

Department of the Navy

Human Factors Engineering Branch
Code 1226
Pacific Missile Test Center
Point Mugu, CA 93042

Dean of the Academic Departments
U.S. Naval Academy
Annapolis, MD 21402

Dr. W. Moroney
Human Factors Section
Systems Engineering Test
Directorate
U.S. Naval Air Test Center
Patuxent River, MD 20670

CDR C. Hutchins
Code 55
Naval Postgraduate School
Monterey, CA 93940

Department of the Army

Mr. J. Barber
HQS, Department of the Army
DAPE-MBR
Washington, D.C. 20310

Dr. Edgar M. Johnson
Technical Director
U.S. Army Research Institute
5001 Eisenhower Avenue
Alexandria, VA 22333

Director, Organizations and
Systems Research Laboratory
U.S. Army Research Institute
5001 Eisenhower Avenue
Alexandria, VA 22333

Technical Director
U.S. Army Human Engineering Labs
Aberdeen Proving Ground, MD 21005

Department of the Air Force

U.S. Air Force Office of Scientific
Research
Life Sciences Directorate, NL
Bolling Air Force Base
Washington, D.C. 20332

Department of the Air Force

AFHRL/LRS TDC
Attn: Susan Ewing
Wright-Patterson AFB, OH 45433

Chief, Systems Engineering Branch
Human Engineering Division
USAF AMRL/HES
Wright-Patterson AFB, OH 45433

Dr. Earl Alluisi
Chief Scientist
AFHRL/CCN
Brooks Air Force Base, TX 78235

Foreign Addresses

Dr. Kenneth Gardner
Applied Psychology Unit
Admiralty Marine Technology
Establishment
Teddington, Middlesex TW11 OLN
England

Director, Human Factors Wing
Defence & Civil Institute of
Environmental Medicine
Post Office Box 2000
Downsview, Ontario M3M 3B9
Canada

Dr. A.D. Baddeley
Director, Applied Psychology Unit
Medical Research Council
15 Chaucer Road
Cambridge, CB2 2EF England

Professor Brian Shackel
Department of Human Sciences
University of Technology
Loughborough, Leicestershire
England LE11 3TU

Other Government Agencies

Defense Technical Information Center
Cameron Station, Bldg. 5
Alexandria, VA 22314 (12 copies)

Dr. C. Kelly
Director, System Sciences Office
Defense Advanced Research Projects
Agency
1400 Wilson Blvd.
Arlington, VA 22209

October 1983

Other Government Agencies

Dr. M. Montemerlo
Human Factors & Simulation
Technology, RTE-6
NASA HQS
Washington, D.C. 20546

Other Organizations

Dr. Jesse Orlansky
Institute for Defense Analyses
1801 N. Beauregard Street
Alexandria, VA 22311

Dr. Robert T. Hennessy
NAS - National Research Council (COHF)
2101 Constitution Avenue, N.W.
Washington, D.C. 20418

Dr. Amos Freedy
Perceptronics, Inc.
6271 Variel Avenue
Woodland Hills, CA 91364

Dr. Deborah Boehm-Davis
General Electric Company
Information Systems Programs
1755 Jefferson Davis Highway
Arlington, VA 22202

Dr. James H. Howard, Jr.
Department of Psychology
Catholic University
Washington, D.C. 20064

Mr. Edward M. Connelly
Performance Measurement
Associates, Inc.
410 Pine Street, S.E.
Suite 300
Vienna, VA 22180

Dr. Marvin Cohen
Decision Science Consortium
Suite 721
7700 Leesburg Pike
Falls Church, VA 22043

Dr. William B. Rouse
School of Industrial and Systems
Engineering
Georgia Institute of Technology
Atlanta, GA 30332

Other Organizations

Dr. Richard Pew
Bolt Beranek & Newman, Inc.
50 Moulton Street
Cambridge, MA 02238

END

FILMED

02-84

DTIC